Archiving pushed Inferences from Sensor Data Streams

Jörg Brunsmann

Faculty of Mathematics and Computer Science, Distance University of Hagen
D-58097 Hagen, Germany

Abstract. Although pervasively deployed, sensors are currently neither highly interconnected nor very intelligent, since they do not know each other and produce only raw data streams. This lack of interoperability and high-level reasoning capabilities are major obstacles for exploiting the full potential of sensor data streams. Since interoperability and reasoning processes require a common understanding, RDF based linked sensor data is used in the semantic sensor web to articulate the meaning of sensor data. This paper shows how to derive higher levels of streamed sensor data understanding by constructing reasoning knowledge with SPARQL. In addition, it is demonstrated how to push these inferences to interested clients in different application domains like social media streaming, weather observation and intelligent product lifecycle maintenance. Finally, the paper describes how real-time pushing of inferences enables provenance tracking and how archiving of inferred events could support further decision making processes.

1 Introduction

Since centuries everyone get used to sensors in the form of classical thermometers that measure temperatures. Nowadays, more sophisticated sensors monitor all kinds of physical world phenomena and they are pervasively deployed, e.g. in cars as acceleration detector, at the door as movement alarm, in the office as smoke detector and in watches as health monitoring sensors. Some of these sensors are enhanced with spatial, temporal and concept annotations to enable high-level reasoning and these sensors are integrated into the semantic sensor web as they are accessible via the internet. Since sensor networks provide a digital interface to real world phenomena, they are part of the i*nternet of things* [7] paradigm. Unfortunately, these sensors are not very intelligent by themselves and are not interconnected, because they produce only raw time series data and the process of combining multiple sensors to identify high-level inferences is not yet very common. In addition, the delivery of sensor inference streams is based on the pull model where it is necessary to actively monitor the streams. This paper describes how to lift raw sensor data to more expressive layers that can be understood both by machines and humans and it describes how to push relevant inferences to interested parties for further processing.

Therefore, this paper combines the following different technologies to form a sensor data processing network that creates, pushes and archives inferences from sensor data streams:

Brunsmann J..
Archiving pushed Inferences from Sensor Data Streams.
DOI: 10.5220/0003116000380046
In *Proceedings of the International Workshop on Semantic Sensor Web* (SSW-2010), pages 38-46
ISBN: 978-989-8425-33-1

1. Executing continuous SPARQL queries [3] over streamed sensor data.
2. Deriving high-level inferences [20] from raw stream data with SPARQL
3. Pushing [13] these inferences to interested clients with sparqlPuSH.
4. Create an archive that accepts and manages stream inferences.
5. Preserving RDF based inferences during ontology evolution.

The remainder of the paper is structured as follows. The next section provides a characterization of sensor networks, semantic sensor data streams, data stream management systems and describes the sensor data pyramid containing different layers of data relationships. Section 3 presents a system architecture that enables generating, pushing and archiving annotated sensor data along the different levels of the sensor data pyramid and applies this architecture to different application domains. The last section concludes with a description of future work.

2 Sensors

A *sensor* is a device with small memory, reduced computing capacities and limited power supply that measures an observable physical quantity or environmental condition like temperature (thermal), sound (acoustic), vibration (seismic), visual light, infrared light, pressure, magnetism, radar, pollutants or motion (e.g. traffic). A sensor monitors a *feature* of interest (e.g. a product or lake) and reports *observations* as value of some *property* representing a quality of a feature obtained using a specified procedure. Sensors vary in terms of the sampling frequency and the amount of data they produce. For example, a thermometer may output a single byte and may be sampled every few seconds whereas a video camera that assists with reversing may output several megabytes per second. A *sensor data stream* is a continuous and timely ordered sequence of observations. A *sensor network* consists of multiple spatially distributed autonomous sensor devices. A sensor network allows that many sensors cooperate and interact with each other. In the *semantic sensor web*, measured sensor data is described by knowledge representations languages like the Resource Description Framework (RDF) [15]. In order to express such semantics, concepts from widely-used vocabularies and data from the *Linked Open Data* [5] cloud is used to allow machine interpretation [2]. Special *Data Stream Management Systems* (DSMS) connect to and monitor one or more stream sources. A DSMS will not archive permanently generated stream data. Such streamed data will rather be processed directly by executing *continuous queries* which are used to produce alarms if some events occur. In addition, a DSMS is used to build aggregated and semantic annotated data from raw stream data. To do so, a DSMS must lift raw sensor data to higher layers of understanding to support both human and machine interpretation. The *sensor data pyramid* is a model that illustrates these different layers.

2.1 Sensor Data Pyramid

The sensor data pyramid [17] that is depicted in figure 1 is a concept for modeling relationships between sensor data, information, knowledge and wisdom.

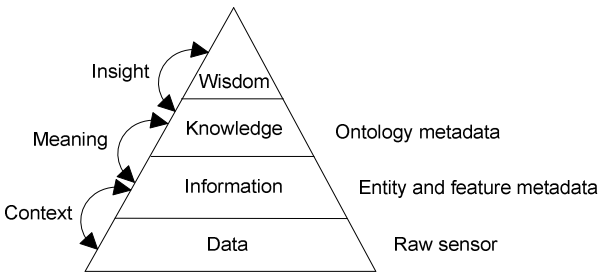

Fig. 1. The sensor data pyramid includes four different layers including raw sensor data, feature metadata and ontology metadata.

In the pyramid, data and associated concepts build information. If meaning is attached to information (e.g. semantic annotation) then knowledge is created. Finally, a human being can have insight into knowledge to gain wisdom. Data at the lower levels will arrive with higher frequency than on higher levels and higher layers have more expressiveness than lower layers.

2.2 Streams of Stream Inferences

Raw stream data that is lifted to higher expressive layers (e.g. via ontology annotations) build new streams (e.g. RDF streams) that serves as input for other consumers.

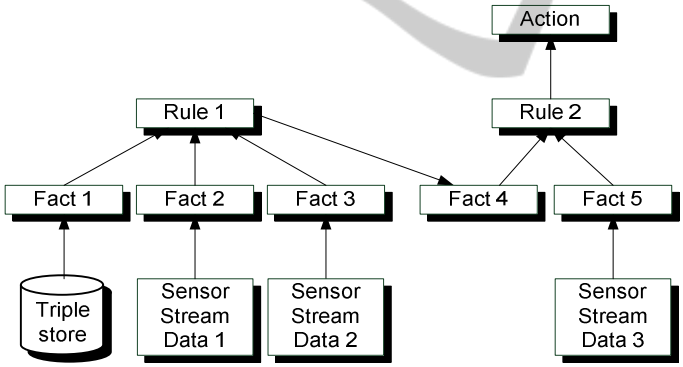

Fig. 2. Raw stream data is lifted to more expressive layers which is then used as input for other stream consumers.

For example see figure 2: rule 1 reads facts (triples) from two different sensors and a database (triple store) to create new facts. These newly created facts (fact 4) are the input for another rule 2 that itself leads to an action.

2.3 Stream Inference Pushing and Archiving Architecture

The sparqlPusH approach [13] broadcasts changes made to RDF data in real-time to

clients as Atom or RSS feeds. The system architecture of sparqlPuSH consists of publishers, hubs and subscribers that can be combined to form a network of multiple actors. The publisher is the source of updates and the hub pushes the newly published data to multiple interested clients that have previously been registered. This architecture can be extended with regard to the data (inferences rather than updates) that is broadcasted. In such a *stream inference pushing and archiving architecture* the workflow is executed in two different steps:

Registration. First, the client registers the inference query at the endpoint of the publisher. The publisher redirects the request to the hub and the client registers for a specific feed published by the hub.

Broadcasting. Second, the publisher executes continuous queries over streams and the publisher executes the sensor data pyramid loop (see below). If registered inference queries generate new results, a notification is sent to the hub. The hub itself broadcasts these inferences to registered feed clients.

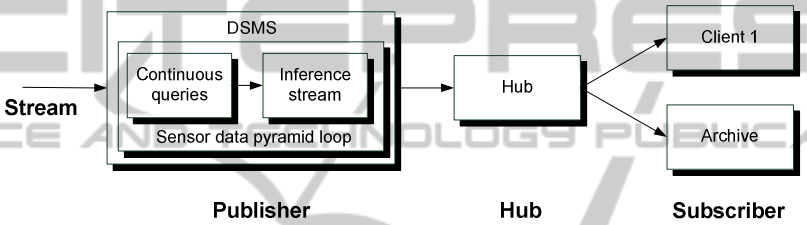

Fig. 3. The broadcasting of inference streams to subscribers after the subscribers has been registered at the hub.

The DSMS handles large amounts of streaming data and executes an endless sensor data pyramid processing loop (figure 3). While executing this loop, the DSMS pushes new inferences to well known hubs which broadcast these inferences to interested clients. Since the DSMS does not archive all data, one of the clients includes a long-term archive that archives and preserves important facts.

Sensor Data Pyramid Loop. The DSMS execute a sensor data pyramid loop that includes reading, analyzing, annotating, reasoning and pushing of stream data. This loop contains the execution of continuous queries like C-SPARQL [3], the generation of inferences and the pushing of newly annotated stream data. In other words, each time new stream data arrives at the DSMS, rules (inference queries) are executed. The SPARQL CONSTRUCT pattern allows expressing such rules because the WHERE clause is the condition that must be fulfilled by current available facts and the CONSTRUCT pattern allows to express the reasoned facts. These reasoned facts are then being used as stream input for other rules. The DSMS will push the inferences to a hub which broadcasts the inferences to clients. One potential client is a long-term archive that archives specific observation events.

The endless sensor data pyramid loop is executed in the DSMS. The loop reads multiple sensor streams, mine and correlate sensor data, annotate one or more streams with semantics, pushes these streams, reason over newly generated facts and pushes potential new generated inferences

```
while true do
  readSensorStreams();
  mineData();
  annotateSensorStream();
  pushSemanticStream();
  infer();
  pushInferenceStream();
end
```

Inferences for Decision Making Processes. The archive client collects all inferences and their provenance. Once all this data has been collected, it can be used by a human to make decisions which can be assisted by machine readable rules.

Preserving Stream Inferences. The inference stream consists of RDF statements that use a specific version of an ontology. These vocabularies evolve over time and archived inferences are only consistent with a specific ontology version. The archive must be able to handle this ontology evolution (threat of semantic obsolescence) by migrating inference instances via ontology mapping. In addition, an inference archive must also have data integration facilities to cope with the heterogeneity of ontologies that describe sensors and sensor data streams.

3 Example Use Cases

In this section several example use cases from different application domains (social media, weather observation and engineering) are described in more detail.

3.1 Social Media Data

Although semantic streams originating from social media do not monitor physical phenomena, they can also be used to infer new knowledge. Microblogging sites like *Twitter* generate streams of short messages. Unfortunately, these streams are not annotated with semantics. Therefore, services described in [18] and [10] and [21] or the ShreddedTweet [19] service can be used to enhance twitter messages to include RDF triples for "*hashtags*". Those RDF triples can then be used to infer knowledge (interests, activity profile and social network analysis) about message posters.

3.2 Weather Observations

One example for the combination of multiple sensors in weather observation is a temperature sensor and a wind speed sensor. Both sensors can be used to calculate the wind chill factor. Another example which is described below makes use of a rain and temperature sensor to derive higher-level knowledge about occurrences of potential freezing rain. In this scenario, the following RDF data (in N3 representation) is created after reading raw stream data. Here, a sensor delivered a concrete value for rain of 0.2 cm in the last hour and a current temperature of minus 1.1 degrees.

After reading the data streams, the following data is attached with semantic information and pushed as new stream (to save space, the prefixes have been left out).

```
ex:obsveration a obs:WheatherObservation ;
      obs:degreesCelsius "-1.1"^^xsd:double ;
      obs:rain "0.2"^^xsd:double ;
      obs:time "2010-06-20 22:30:03.0" .
```

These facts can be used to infer new facts. That means, after producing the above semantic stream, the following SPARQL CONSTRUCT query can be used to infer new facts.

Rule to infer freezing rain which occurs if the temperature is below zero and if rain has fallen. The CONSTRUCT pattern creates new facts if the condition in the WHERE clause is fulfilled

```
CONSTRUCT { obs:freezingrain prov:where geo:hagen .
            obs:freezingrain prov:why ex:reason .
            obs:freezingrain prov:when ?time .
            ex:reason obs:degreesCelsius ?temp .
            ex:reason obs:rain ?rain . }
WHERE { ?s obs:degreesCelsius ?temp .
        ?s obs:rain ?rain .
        ?s obs:time ?time .
        FILTER ( ?rain > 0 && ?temp < 0) }
```


If the condition in the WHERE clause applies, new facts will be created that include the *inference provenance.* According to [9] the following metadata are of interest while creating observations: *Where* is the geographical location of the captured data? *When* did the observation begin and when did it end? *What* was measured? *How* the observation was executed (instruments)? *Who* is the originator of the data? To describe this provenance, the W7 provenance model [14] can be used to attach provenance to specific events. The following code shows the inferred facts.
A freezing rain inference attached with provenance information produced from a SPARQL CONSTRUCT rule

```
ex:reason
      obs:degreesCelsius "-1.2"^^xsd:double ;
      obs:rain "1.2"^^xsd:double .
obs:freezingrain
      prov:when "2010-06-20 22:30:03.0" ;
      prov:where geo:hagen ;
      prov:why ex:reason .
```

3.3 Sensors in Product Lifecycle Management

Product lifecycle management (PLM) systems allow the management of product related information for the complete lifecycle of a product from its conception through design and manufacturing to service and disposal. During product operation and service valuable data can be obtained from products by using sensors [16] [1]. Sensors enable to continuously monitor the conditions of a product. If such data is broadcasted, it can be collected and shared as PLM data among lifecycle actors. Data obtained from sensors during product operation can be used to detect needs for main-

tenance, service or repair in advance. After broadcasting such conditions, spare parts can be ordered and service personnel can be scheduled. In addition, sensor data help to proactively identify and report conditions before a failure occurs. Other functional aspects of collecting product data from sensors include:

Product Tracing. Products may be equipped with low cost radio frequency identification (RFID) and wireless communication that enable automated tracking and tracing the origin, location, movements, physical properties, environmental conditions and usage history of a product. The tracing of products help to fight counterfeit products.

Product Diagnostics. Modern vehicles contain multiples sensors that monitor various aspects of the engine's operation, such as thermometers or fuel flow rate which are not only interesting for the driver but also for the service company. Such data can be broadcasted in real-time or collected in a memory to be passed later to a remote node. In addition, product diagnostics data improve the way that products are recycled when they arrive to their end-of-life.

Product Manufacturing Processes. Sensor data can help to improve existing business processes and help to adapt processes in real time. By giving real-time product status information, manufacturing procedures can be improved. For example, the car sensor data can be broadcasted to the vehicle manufacturer. If the vehicle manufacturer could collect such real-use information from a big amount of cars, maintenance scheduling can be planned more efficiently.

Another notion of sensor enriched products is *intelligent products* [11] that collect usage information and react on it proactively. Intelligent products can be divided into three categories: an intelligent product should at least be able to manage its own information given by sensors, RFID-readers (*information handling aspect*). A more intelligent product is a product which can report (*problem notification aspect*) when there is a problem (e.g. temperature is too high). Finally, the most intelligent product is the product which can completely manage its own life (*decision making aspect*).

4 Conclusions and Outlook

This paper presented a system architecture that combines different technologies in order to enable sensor stream processing together with ontology representations and standard SPARQL inference formalisms to derive high-level understandings of raw streamed sensor data. It has been described how reasoning is achieved in the semantic sensor web with standard technologies like RDF based linked data and continuous inference queries based on the SPARQL CONSTRUCT pattern. In addition, this paper described how inferences are pushed to interested clients via sparqlPuSH technology. Then, three different example use-cases where described in more detail from social media, weather observation and engineering.

Future research will be targeted to the analysis of systems with similar capabilities and the evaluation of possibilities to integrate the described solutions into existing systems. Further investigations include evaluation of other rule languages like RIF (Rule Interchange Format) or XChange. [6], N3Logic [4] or ECA-Rules [12]. In addi-

tion, research will be executed how decision rules can be formulated and executed that help decision making [22]. Further investigations will also include the integration of ontology mappings into archive systems the preserve ontology based stream inferences. Finally, while several ontologies exists that describe sensors and sensor observation [8], domain specific sensor knowledge ontologies (e.g. product monitoring) are missing. For example, a common vocabulary for product lifecycle metadata is required which is necessary to support exchanging of PLM data during whole product lifecycle.

Acknowledgements

This paper is supported by the European Union in the 7th Framework within the IP SHAMAN.

References

1. Anke, J., Främling, K.: Distributed Decision Support in a PLM scenario. Proceedings of Product Data Technology Europe 14th Symposium (2005)
2. Barbieri D. F., Valle E.D.: A Proposal for Publishing Data Streams as Linked Data. Linked Data on the Web Workshop (2010)
3. Barbieri D. F., Braga D., Ceri S., Della Valle E., Grossniklaus M.: Continuous queries and real-time analysis of social semantic data with c-sparql. Proceedings of Social Data on the Web Workshop at the 8th International Semantic Web Conference (2009)
4. Berners-Lee, T., Connolly D., Kagal L., Hendler J., Scharf Y.: N3Logic: A Logical Framework for the World Wide Web. Journal of Theory and Practice of Logic Programming (TPLP), Special Issue on Logic Programming and the Web (2008)
5. Bizer C., Heath T. Berners-Lee T.: Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems (2009)
6. Bry, F., Eckert, M., Patranjan, P.-L., Romanenko, I.: Realizing Business Processes with ECA Rules: Benefits, Challenges, Limits. Proc. Int. Workshop on Principles and Practice of Semantic Web. LNCS, Springer, Heidelberg (2006)
7. Christin D., Reinhardt A., Mogre P. S., Steinmetz R.: Wireless Sensor Networks and the Internet of Things: Selected Challenges. Proceedings of the 8th GI/ITG KuVS Fachgespräch Drahtlose Sensornetze (2009)
8. Compton, M., Henson, C., Lefort, L., Neuhaus, H.: A survey of the semantic specification of sensors. Technical report (2009)
9. Hayes, J., O'Conor, E., Cleary, J., Kolar, H., McCarthy, R., Tynan, R., O'Hare, R., Smeaton, A., O'Connor, N., Diamond, D.: Views From the Coalface: Chemo-Sensors, Sensor Networks and the Semantic Sensor Web. International Workshop on the Semantic Sensor Web (2009)
10. Hepp, M.: HyperTwitter: Collaborative Knowledge Engineering via Twitter Messages. Technical Report (2010)
11. Meyer, G.G., Främling, K. & Holmström, J.: Intelligent Products: A survey. Computers in Industry, 60 (2009)
12. Papamarkos G., Poulovassilis A., Wood P.T.: Event-Condition-Action Rules on RDF Metadata in P2P Environments. In Proc. 2nd Workshop on Metadata Management in Grid and P2P Systems (MMGPS): Models, Services and Architectures, London (2004)

13. Passant, A., Mendes, P. N.: sparqlPuSH: Proactive notification of data updates in RDF stores using PubSubHubbub. 6th Workshop on Scripting for the Semantic Web (2010)
14. Ram S. Liu J.: Understanding the Semantics of Data Provenance to Support Active Conceptual Modeling. Proceedings of the Active Conceptual Modeling of Learning Workshop (2006)
15. Rodriguez, A., McGrath, R., Liu, Y., Myers, J.: Semantic Management of Streaming Data. Proc. Intl. Workshop on Semantic Sensor Networks (2009)
16. Seitz C., Legat C., Neidig J.: Embedding Semantic Product Memories in the Web of Things. First International Workshop on the Web of Things (2010)
17. Sheth A., Henson C., Sahoo S. S. Semantic Sensor Web. ARC Research Network on Intelligent Sensors, Sensor Networks and Information Processing – ISSNIP, Melbourne (2008)
18. Shinavier J.: Real-time #SemanticWeb in <= 140 chars. Proceedings of the Third Workshop on Linked Data on the Web (2010)
19. Shredded Tweet: http://pegasus.chem.soton.ac.uk/
20. Stuckenschmidt H., Ceri S., Valle, E.D., van Harmelen F.: Towards Expressive Stream Reasoning. Proceedings of the Dagstuhl Seminar on Semantic Aspects of Sensor Networks (2010)
21. Twarql – Software implementation realizing the Linked Open Social Signals vision: http://wiki.knoesis.org/index.php/Linked_Open_Social_Signals
22. Zangiacomi A., Fornasiero R.: Modelling decision support systems for Middle-of-life in product lifecycle management. 14th International Conference on Concurrent Enterprising, Lisbon, Portugal (2008)